

1 **VALIDATION OF NETWORK COMMUNICATION TUNNELS**

2 **FIELD OF THE INVENTION**

3 This invention is directed to the field of computer
4 networks. It is more particularly directed to network
5 devices that transform packets traveling in an IP
6 network in various manners.

7 **BACKGROUND OF THE INVENTION**

8 The Internet Protocol (herein referred to as IP) is a
9 preferred communication mechanism used in the Internet
10 and many enterprise networks. The original IP
11 communication was designed so that the network would
12 deliver the IP packets created by a source machine to a
13 destination machine without any significant
14 modifications to the contents of the IP packet. Some
15 minor modifications were made to the contents of the
16 header of an IP Packets during the course of normal
17 processing, but most of the fields of the IP header and
18 the payload were left unmodified.

19 In order to meet the varying security and scalability
20 needs that arise within the Internet and enterprise IP
21 intranets, it has become common practice to perform
22 different types of transformations on an IP packet.
23 Example of such transformations include the encryption

1 of IP packets, compressing IP payload, encapsulating an
2 IP packet within another IP packet, translating the
3 source/destination address to some other value within
4 the network, etc. Such transformations are most often
5 performed in a transparent fashion, such that an
6 application that is sending the IP packets in the
7 network is often not aware of the fact that such
8 transformations are being done within the network. A
9 transformation may be done by the kernel of an
10 originating machine, or by an intermediary router or
11 firewall in the network. Such transparent
12 transformations are hidden from the receiving
13 application by a reverse transformation which is
14 performed in the kernel of the receiving machine or at
15 another intermediary device close to the receiving
16 machine. These transformations establish logical
17 communication tunnels within an IP network between the
18 devices that perform the transformation and the devices
19 that perform the reverse transformation. The IP
20 communication tunnel could be established on the entire
21 path of an IP packet, when the transformation is done
22 at the source machine and the reverse transformation at
23 the destination machine. An IP communication tunnel
24 could also apply only to part of a route between a pair
25 of communicating machines, with a network device doing
26 the transformation at a point in the path of the
27 packet, and another network device performing the
28 reverse transformation in the path of the packet.

29 An example of a transformation process, is the
30 IP-security protocol using the Encrypted Secure Payload
31 mechanism in a tunnel mode. It enables a router (or

1 source machine) to encrypt an IP packet and put it
2 within a new IP packet header on the source side. This
3 transformation is reversed close to the destination
4 machine by a router and the original IP packet is
5 regenerated. The reverse transformation can also be
6 done by the destination machine kernel itself. Other
7 types of transformations include IP security protocol
8 using the Encrypted Secure Payload in a transport mode,
9 IP security protocol using the Authentication header in
10 a transport mode, IP security protocol using the
11 Authentication Header in a tunnel mode, Generic Routing
12 Encapsulation, Network Address translation etc. These
13 transformations are described in various Internet
14 Request For Comments (RFCs), namely RFC 2401, RFC 2406,
15 RFC 2402, RFC 1701, RFC 1702, RFC 1631, RFC 2766 etc.,
16 These transformations are known to those skilled in the
17 art.

18 *Sub 21* ~~These Internet Network Working Group submissions are~~
19 herein incorporated by reference in entirety. These
20 are available on the Internet at the following Internet
21 URLs:

22 RFC 2401: <http://www.ietf.org/rfc/rfc2401.txt>
23 RFC 2406: <http://www.ietf.org/rfc/rfc2406.txt>
24 RFC 2402: <http://www.ietf.org/rfc/rfc2402.txt>
25 RFC 1701: <http://www.ietf.org/rfc/rfc1701.txt>
26 RFC 1702: <http://www.ietf.org/rfc/rfc1702.txt>
27 RFC 1631: <http://www.ietf.org/rfc/rfc1631.txt>
28 ~~RFC 2766: <http://www.ietf.org/rfc/rfc2766.txt>~~

1 One of the key aspects of these transformations is that
2 they are to be applied transparently to IP packets such
3 that the originating applications are not aware of the
4 existence of the transformation process. The
5 transformation process is usually put into place by
6 configuration and administration of the network devices
7 (servers, firewalls or routers) that perform the
8 transformation. However, in many cases, one needs to
9 validate that the transformation has actually been
10 accomplished, and that the communicating applications
11 are communicating with each other using the
12 transformation.

13 A usual mechanism to ensure that things are set up
14 properly for two machines to communicate on an IP
15 network is to use a program like ping. It sends an IP
16 packet defined according to the Internet ICMP standards
17 from one machine to another, and the latter provides a
18 response to the original ICMP packet. Upon receiving
19 the ICMP packet, the original machine is assured that
20 communication is possible between the two machines.
21 However, when transformation are in place, it is
22 advantageous to have a way to ensure not only that the
23 communication is possible, but that the communication
24 is happening with the right transformations taking
25 place. Communication may be taking place without any
26 transformations, or may be happening with incorrect
27 transformations. In most environments, the
28 transformation should be occurring correctly. It is
29 unacceptable that packets be sent in the clear when
30 they require an IP-sec encryption transformation.
31 Existing schemes for ensuring that there is

1 connectivity between machines can not be used for the
2 purpose of validating communication when
3 transformations are taking place.

4 It would also be advantageous that the validation of
5 the communication be able to be accomplished by an
6 individual operating from one node, or by a management
7 console which is responsible for managing the
8 configuration of multiple machines in an IP network.

9 SUMMARY OF THE INVENTION

10 It is therefore an aspect of the present invention to
11 provide methods and apparatus by which a network user
12 or administrator at a central console can validate that
13 transformations required for communication to another
14 machine have been set up properly.

15 In another aspect of the invention, a network user or
16 administrator uses a method disclosed herein to
17 validate that IP-sec tunnels performing Encrypted
18 Secure Payload transformations in tunnel or transport
19 mode have been established properly between
20 communicating firewalls, routers and/or hosts.

21 In still another aspect of the invention, a network
22 user or administrator uses a method disclosed herein to
23 validate that IP-sec tunnels performing Authentication
24 Header transformations in tunnel or transport mode have
25 been established properly between communicating
26 firewalls, routers and/or hosts.

1 In an alternative embodiment, a network user or
2 administrator uses a method disclosed herein to
3 validate that Generic Routing Encapsulation performing
4 IP in IP encapsulation have been established properly
5 between communicating firewalls, routers and/or hosts.

6 In a further aspect of the invention, a network user or
7 administrator uses a method disclosed herein to
8 validate that network address translation has been
9 established properly between communicating network
10 devices.

11 Other objects and a better understanding of the
12 invention may be realized by referring to the detailed
13 description.

14 **BRIEF DESCRIPTION OF THE DRAWINGS**

15 These and other aspects, features, and advantages of
16 the present invention will become apparent upon further
17 consideration of the following detailed description of
18 the invention when read in conjunction with the drawing
19 figures, in which:

20 Fig. 1 shows an example of an environment having
21 multiple network devices deployed in an IP network
22 which use transformations among various points within
23 the network;

1 Fig. 2 shows an instance of the environment shown in
2 Figure 1, in which the specific transformation used is
3 IP-sec based encryption using the Encrypted Secure
4 Payload or the Authenticated Header mechanisms;

5 Fig. 3 shows an example deployment of validation
6 mechanisms described in this invention in the
7 environment described by Fig. 1 and Fig. 2;

8 Fig. 4 shows an example of a flowchart of a method used
9 for validation mechanism described in this invention by
10 a program running on a machine that initiates the
11 validation scheme, i. e. the client machine for the
12 validation scheme;

13 Fig. 5 shows an example of a flowchart of a method used
14 for validation mechanism described in this invention by
15 the server daemon program running on a machine that
16 participates in the validation scheme;

17 Fig. 6 shows an example of a flowchart of an
18 alternative method used for a validation mechanism
19 described in this invention by the server daemon
20 program running on a machine that participates in the
21 validation scheme;

22 Fig. 7 shows an example of a flowchart of an
23 alternative embodiment used for a validation mechanism
24 described in this invention by a client machine;

25 Fig. 8 shows an example of a flowchart of a method used
26 for a validation mechanism described in this invention

1 by the server daemon program running on a machine that
2 participates in the validation scheme when the client
3 program uses the method shown in Fig. 7;

4 Fig. 9 shows an example of a flowchart of a method for
5 performing a validation step utilizing packets created
6 with a known time-to-live within the network;

7 Fig. 10 shows an example of a flowchart of an
8 alternative method for performing a validation step
9 utilizing access to network device driver sockets;

10 Fig. 11 shows an example of a flowchart of an
11 alternative method for performing a validation step
12 utilizing creation of dummy interfaces to route packets
13 within the network;

14 Fig. 12 shows an example apparatus that implements a
15 validation methods in accordance with the present
16 invention;

17 Fig. 13 shows an alternate apparatus that implements a
18 validation method in accordance with the present
19 invention;

20 Fig. 14 shows an example environment in which the
21 validation mechanism is used to check partial route
22 transformation when a transformation only occurs on a
23 part of the route used for communication among
24 different machines in accordance with the present
25 invention; and

1 Fig. 15 shows an example of a flowchart of a method for
2 validation partial route transformations in the
3 environment shown by Fig. 14 in accordance with the
4 present invention.

5 DESCRIPTION OF THE INVENTION

6 The present invention provides methods and apparatus
7 for validating transformations that occur in an IP
8 network. It is generally directed to servers, routers,
9 firewalls and other network devices that deploy
10 techniques to transform packets traveling in an IP
11 network in various manners, e.g. encrypting them for
12 secure transmission, compressing the packets for
13 reducing bandwidth usage, or translating network
14 addresses. A typical environment in which
15 transformations occur is illustrated in Figure 1. It
16 shows an example of an environment with multiple
17 network devices deployed in an IP network which use
18 transformations among various points within the
19 network.

20 Figure 1 shows a scenario wherein the IP subnetworks
21 (105, 111) and hosts 103, 109, 115, 117 are connected
22 to a core IP network 107. An example of such a core IP
23 network is the Internet, a collection of networks using
24 the IP protocol. Hosts can be connected directly to
25 the core IP network 107, as exemplified by hosts 103
26 and 117, or they could be connected via intermediary
27 routers and subnetworks. In the figure, hosts 109 and
28 115 connect to the core network through intermediary

1 routers and subnets. Host 109 is connected to the
2 subnetwork 111, which connects to the core network 107
3 through the router 101. Host 115 is connected to the
4 subnetwork 105 which connects to the core network via
5 the router 113. IP packets exchanged between
6 subnetworks or hosts connected directly to the Internet
7 are transformed at transformation points just before
8 they enter and as soon as they leave the core IP
9 network 107.

10 In Figure 1, transformations can be done at hosts 103
11 and 117, as well as at routers 101 and 113. Examples
12 of such transformations include compression functions
13 to exploit the shared bandwidth of the Internet more
14 effectively, tunneling (e. g. GRE / Network Address
15 Translation) to hide network internal addresses against
16 the Internet, or encryption or authentication functions
17 (e. g. IP-sec) to protect data against unauthorized
18 disclosure or unnoticed unauthorized change (forging).
19 The transformation deployed at the receiving side
20 usually inverts the transformation applied at the
21 sender side.

22 If the transformations for a traffic flow between hosts
23 109 and 115 are done at network routers 101 and 114,
24 they are transparent to the hosts. In this case, the
25 hosts benefit from these transformations without having
26 to care about their implementation and maintenance.
27 The transformation for a traffic flow between hosts 103
28 and 117 occurs at the hosts themselves, but are done in
29 a manner so as to be transparent to the applications
30 running on these hosts. Thus, in this embodiment,

1 various IP communication tunnels are identified. Each
2 IP communication tunnel is defined by a pair of network
3 devices which do a transformation and inverse
4 transformation of packets traveling on the network. An
5 example of such a communication tunnel is established
6 between the network routers 101 and 114, while another
7 example would be established between the hosts 103 and
8 117.

9 Figure 2 shows an instance of the environment shown in
10 Figure 1, wherein the specific transformation used is
11 IP-sec based encryption using the Encapsulating
12 Security Payload or the Authenticated Header
13 mechanisms. In this instance, IP-sec (as described in
14 Internet RFC - 2401) represents both a set of
15 transformations to protect IP packets sent over
16 distrusted subnetworks and mechanisms to negotiate the
17 mode (tunnel or transport mode, ESP or AH protocol) and
18 the actually employed transformations (e. g. triple
19 des, des, hmac-md5, hmac-sha1). The negotiation is
20 usually implemented by user processes based on the
21 Internet Key Exchange protocol (IKE as defined in
22 Internet RFC 2409). The IKE communication is based on
23 the ISAKMP message exchange protocols defined in RFC
24 2408.

25 The scenario shown in Figure 2 parallels that of Figure
26 1, with a set of trusted IP subnetworks (205, 211) and
27 trusted hosts 203, 209, 215, 217 connected to a an
28 untrusted IP network 207. An example of such a
29 untrusted IP network is the Internet. Examples of
30 trusted subnetwork and servers are the networks and

1 machines operated by an enterprise connected to the
2 Internet. The trust relationship is from the point of
3 the view of the enterprise deploying the subnetworks
4 205, 211 and the hosts 203, 209, 215, etc. Some
5 trusted hosts can be connected directly to the
6 untrusted IP network 207 , as exemplified by hosts 203
7 and 217, or they could be connected via intermediary
8 routers, firewalls and subnetworks. In the figure,
9 hosts 209 and 215 connect to the core network through
10 intermediary routers and subnets. Host 209 is
11 connected to the subnetwork 211, which connects to the
12 core network 207 through the router 201. Host 215 is
13 connected to the subnetwork 205 which connects to the
14 core network via the router 213. IP packets exchanged
15 between subnetworks or hosts connected directly to the
16 Internet are encrypted or authenticated just before
17 they enter and as soon as they leave the untrusted IP
18 network 207.

19 In the embodiment of Figure 2, such encryption or
20 authentication using the IP-security ESP or AH
21 protocols is done at hosts 203 and 217, as well as at
22 routers 201 and 213. The decryption of encrypted
23 packets arriving from the untrusted network 207 is done
24 for a traffic flow between hosts 209 and 215 are done
25 at network routers 201 and 214, and is transparent for
26 the hosts. The encryption/authentication for a traffic
27 flow between hosts 203 and 217 occurs at the hosts
28 themselves, but are done in a manner so as to be
29 transparent to the applications running on these hosts.

30 Before packets can be exchanged between the subnetworks
31 (cf. 201) over the Internet (cf. 203), related IKE

1 processes as part of the IP-sec routers (cf. 205) will
2 negotiate the transformations to be deployed.
3 Afterwards, IP packets that leave the IP-sec routers to
4 enter the Internet will be encapsulated using the
5 Encapsulating Security Payload or the Authentication
6 Header protocol according to the negotiated transforms
7 and modes. When the protected packets arrive at the
8 peer IP-sec router, the inverse transforms are applied
9 and the resulting clear text IP packets are sent into
10 the peer subnetwork. If the IP-sec functions are
11 applied within a host 207, the packets are encapsulated
12 and decapsulated within the hosts before they are sent
13 via the network interface into the IP network.
14 Generally, packets are protected on the path between
15 peer IP-sec transformation functions. As the
16 transforms are applied pairwise, they can be verified
17 only in between the transformation. In the present
18 embodiment, packets are inspected as they appear in
19 between peer transformation points.

20 In order to validate the transformations that are
21 occurring in the network, we augment the traditional
22 packet processing in machines with additional
23 functions. Figure 3 shows the structure for a machine
24 that is running on the network that can deploy the
25 present invention. The machine 301 could be a end-host
26 (e. g. 103 in Figure 1) that does the transformation,
27 a router (e. g. 113 in Figure 1) that does the
28 transformation. It can also be a machine that does not
29 do the transformation (e. g. 109 in Figure 1), but
30 relies on some other device to do the transformation in
31 the network. Any device 301 implementing this

1 validation method includes two software components, a
2 validation daemon 305 and a validation client 303. In
3 some embodiments of this invention, the validation
4 daemon 305 need not be present as a separate module.
5 In these embodiment the functionality of the validation
6 daemon is already provided by traditional IP services.
7 The validation client 303 initiates the validation
8 scheme, and the validation daemon 305 which responds to
9 messages generated by a validation client running on a
10 different machines. Both the validation client 303 and
11 the validation daemon 305 depend on the packet
12 transmission capability available in the box 301 to
13 communicate with the other devices in the network.

14 In an embodiment, when it is desired to validate the
15 transformations that are happening on a packet flow
16 between a pair of machines, the validation client is
17 invoked on one of the machines. This machine is the
18 originator of the validation process. This validation
19 process validates that the transformations are
20 occurring properly on the IP communication tunnel
21 established between the two machines. When the
22 validation client 301 on the originator initiates a
23 sequence of message to verify that transformations are
24 occurring on a set of packets being exchanged between
25 itself and another participant in the tunnel, the first
26 step is to verify that the expected transformation is
27 applied on packets leaving the local machine to the
28 server. If this test is successful, the validation
29 client 303 on the originator machine sends a requests
30 to the validation daemon 305 on the participant machine
31 to make a similar validation on its end. An exchange

1 of messages is made to ensure that the communication is
2 feasible between the machines on the tunnel. If the
3 expected transformations are applied in both
4 directions, the validation process is said to be
5 completed successfully.

6 The present invention considers various transformations
7 like IP-sec in tunnel and transport mode, GRE, general
8 compression, Masquerading, PPP tunneling, etc. It
9 includes not only the procedures for the local tests
10 but also the protocol for the communication between
11 client and server including message types, message
12 fields, and the tasks to be applied on sending and
13 receiving respective messages (communication protocol).

14 Figure 4 shows an example flowchart that illustrates a
15 method that used for a validation mechanism by the
16 validation client 303. The flowchart is entered in
17 step 401 whenever the validation client on a validation
18 process originator machine is invoked. Such an
19 invocation can be done by an operator, or by a script
20 in the absence of an operator. In step 403, the method
21 validates that the transformations are occurring as
22 expected on packets that are originating from it.
23 Detailed descriptions of various techniques that can be
24 used to perform this step are described subsequently in
25 regard to Figures 9, 10 and 11.

26 If the validate transformation step 403 fails, the
27 validation process stops with failure in step 411. If
28 the validate transformation step 403 succeeds,
29 indicating that the transformation is happening as

1 indicated, the send message procedure in step 405 is
2 executed. This involves sending a message to the other
3 participant in the tunnel. After the send message
4 step, the originator waits in step 407 for a
5 predetermined amount of time T to receive a response
6 from the other participant. If no such message is
7 received in step 407, the validation process stops with
8 failure in step 411. If a message is received in step
9 407, the originator in step 409 validates that the
10 response received is valid and as expected according to
11 the transformation. If the response from the other
12 participant is found to be valid, the validation step
13 409 succeeds, and the method declares the validation
14 process to be successful and terminates in step 413.
15 If at any of the steps in the method, the process does
16 not succeed, the method declares the configuration to
17 be invalid and terminates in step 411.

18 Figure 5 shows an example flowchart that illustrates a
19 method used for a validation mechanism described in
20 this invention by the validation daemon 305 running on
21 a machine that participates in the validation scheme.
22 The flowchart is entered in step 501 upon the
23 initialization of the validation daemon 305. Such an
24 initialization may be done when a machine is powered on
25 or booted up, or it can be done by means of an explicit
26 command when the machine is powered on. The validation
27 daemon waits for messages from the client which is
28 participating in the validation scheme (step 503).
29 Once it receives a message, it executes a series of
30 validation steps, generates a response and sends it
31 back to the client (step 505). The validation daemon

1 continues in the loop shown in Figure 5 until it is
2 stopped.

3 Figure 6 shows an example flowchart that illustrates an
4 alternative method used for a validation mechanism by
5 the server daemon program running on a machine that
6 participates in the validation scheme. The server
7 waits for messages from the client which is
8 participating in the validation scheme (step 603). It
9 then validates its own transformation (step 605).
10 This can be done by performing one of the validation
11 methods shown subsequently in Figures 9, 10 and 11. If
12 the transformation is correct, it generates a positive
13 response and sends it back to the client (step 607).

14 Figure 7 shows an example of a flowchart that
15 illustrates an alternative embodiment of a method
16 useful for a validation mechanism by a client machine.
17 The client first validates the transformation locally
18 (step 703). This can be done by performing one of the
19 validation methods as shown in figures 9, 10 and 11.
20 If the transformation is correct, it validates the
21 transformation used by its partner which is the other
22 machine participating in the validation scheme (step
23 705).

24 The validation of the transformation can be done by
25 sending a message to the other end containing a random
26 number. If the other end is able to receive this
27 message, increment the number and perform the correct
28 transformation, we know that the transformation is
29 taking place correctly. Once this check passes, it

1 then sends a message to the other end (step 707) and
2 waits for a response to be received within a period of
3 time (say T seconds) (step 709). If the response is
4 received within T seconds, the client then checks for
5 the validity of the received message (step 711). The
6 response can either be positive or negative based on
7 the transformation checks performed on the server.

8 Figure 8 shows an example of a flowchart that
9 illustrates a method used for a validation mechanism by
10 the validation daemon program running on a machine that
11 participates in the validation scheme when the
12 validation client program uses the method shown in
13 Figure 7. The validation daemon waits for messages
14 received from the client (step 803). It then checks
15 for the validity of the message (step 805) which means
16 checking to see if the message is a validation request.
17 If the message is valid, it validates its own
18 transformation (step 807). This can be done by
19 performing one of the validation methods as shown in
20 figures 9, 10 and 11. It then sends a response to the
21 client notifying it of the validity of the
22 transformation. The response can be either a positive
23 response or a negative response base on the outcome of
24 the transformation check. It then waits for more
25 incoming messages (step 803). On the other hand if the
26 message received from the client was not valid, it
27 check to see if the message is a communication message.
28 If it is, it then generates a response to the sender
29 (step 813). If it is not a communication message, it
30 discards the message (step 815) and returns to waiting
31 for more messages (step 803).

1 Figure 9 shows a flowchart illustrating an example
2 method for an embodiment performing a validation step
3 required in the invention. The method is entered in
4 step 901. The method shown utilizes packets created
5 with a known time-to-live within the network. The
6 client creates a socket and sets the time-to-live (TTL)
7 value for packets outgoing on that socket to be '1'
8 (step 902). It then sends a packet to the server via
9 this socket (step 903) and waits for a response to be
10 received (step 904). This packet reaches the first-hop
11 router towards the server via standard IP routing. At
12 a router, it is standard practice to decrement the TTL
13 of every incoming packet. If the TTL then becomes '0',
14 a ICMP packet is generated by the router and sent back
15 to the originating host. This ICMP packet contains the
16 header and first 8 bytes of the offending packet.
17 Thus, the packet sent by the client with TTL=1 (in step
18 903) is intercepted by the first-hop router and a ICMP
19 message is received by the client (step 904). The
20 client then inspects the header of the original packet
21 contained within the ICMP message. If the header and
22 the succeeding eight bytes demonstrate that the
23 necessary transformation was applied to the packet
24 (step 905), then it is validated that the client is
25 applying a transformation function on outgoing packets
26 to the server (step 906). If the client does not
27 receive a response from the first-hop router within a
28 time interval T or if the returned ICMP message
29 contains a IP header and eight bytes that suggests a
30 transformation function was not applied by the client,

1 then in either case, the validation step fails (step
2 907).

3 Figure 10 shows an example of a flowchart illustration
4 of an alternative method for performing the validation
5 step. The method utilizes access to network device
6 driver sockets. The client creates a control socket to
7 snoop on all packets outgoing through a given network
8 device (step 1002). The client then sends a packet to
9 the server. This packet will then undergo a
10 transformation before being sent towards the server
11 via this network device (step 1003). The client waits
12 for a time interval T to capture this outgoing packet
13 via the control socket (step 1004). Once this packet
14 is captured, its header will be examined to check if a
15 transformation was applied to the outgoing packet (step
16 1005). If so, then this serves to validate that a
17 transformation is being applied to packets sent towards
18 the server. If the packet capture is not successful
19 in step 1004 or if the header information of the
20 captured packet suggests that the transformation was
21 not applied, then the validation step fails.

22 The method shown in Figure 10 has the advantage that it
23 would work even for transformations that modify the
24 Time To Live field within the IP header. An example of
25 such a transformation is IP-security ESP in the tunnel
26 mode, which resets the TTL of the outer header to a
27 predetermined value. For such transformations, the
28 approach shown in Figure 9 would not work. However,
29 the approach shown in Figure 10 would still work.

1 A specific implementation of the method shown in Figure
2 10 is useful in an embodiment of the invention which
3 does not require a validation daemon at the
4 participating nodes. In this embodiment, the
5 originator of the validation process would initially
6 validate that transformations are occurring on packets
7 being sent out from its side of the IP communication
8 tunnel. It then sends an ICMP echo request to the
9 participating tunnel. The ICMP echo requests are
10 created as defined by established IP standards. In
11 response to the echo request, an echo reply is
12 received. The packet capture mechanism shown in Figure
13 10 is used to obtain a copy of the reply before the
14 inverse transformations are applied. It is examined to
15 ensure that the required transformations are occurring
16 on packets originating from the other participant in
17 the tunnel. A copy of the reply is also received after
18 the transformations by the validation client, and it is
19 verified that communication is indeed possible on the
20 IP communication tunnel.

21 Figure 11 shows an example of a flowchart for an
22 alternative method for performing the validation step.
23 The method shown utilizes creation of dummy interfaces
24 to route packets within the network. In this method,
25 the client creates a dummy interface and assigns an
26 address to that interface that is same as the server's
27 address (step 1102). Consequently, any packet sent
28 from the client (step 1103) that is addressed to the
29 server is routed back to the client itself in a
30 loopback fashion, instead of being routed into the
31 network towards the actual server. The client waits a

1 time interval T to receive this packet on its dummy
2 interface (step 1104). If the captured packet header
3 demonstrates that a transformation function was applied
4 by the client, then this serves as a positive
5 validation test. Otherwise, if the outgoing packet is
6 not received on the dummy interface or if the packet
7 header do no demonstrate an application of the
8 transformation function to the packet, then the
9 validation step fails.

10 Figure 12 shows an example of an apparatus that
11 implements the validation methods as described in this
12 invention. A network device 1201 which implements this
13 inventions comprises a transformation validator 1207, a
14 communication validator 1205, and a packet
15 sender/receiver module 1203. The transformation
16 validator 1207 validates that packets being sent out of
17 the device 1201 are transformed properly. The
18 communication validator 1205 validates that the packets
19 being sent to the device 1201 by other participants in
20 the IP communication tunnel are being transformed
21 properly. These two modules perform their actions by
22 implementing the various methods shown in Figures 4
23 through 11. Both modules 1205 and 1207 send packets to
24 other devices within the network using the packet
25 send/receive module 1203.

26 Figure 13 shows an apparatus that implements an
27 alternative embodiment of the validation method as
28 disclosed in this invention. A network device 1309
29 implementing the disclosure comprises a transformation
30 validator 1301, a remote party transformation validator

1 1303, a communication validator 1305, and a packet
2 sender/receiver module 1307. The transformation
3 validator 1301 validates that packets being sent out of
4 the device 1309 are transformed properly. The
5 communication validator 1305 validates that the packets
6 being sent to the device 1309 by other participants in
7 the IP communication tunnel are being transformed
8 properly. The remote party transformation validator
9 1303 is a module that receives requests from other
10 participants in the IP communication tunnel, and
11 validates that local transformations are occurring as
12 expected on the receipt of the messages. It also sends
13 a response back to the participants. These three
14 modules perform their actions by implementing the
15 various methods shown in Figures 4 through 11. All
16 three modules 1301, 1303 and 1305 send packets to other
17 devices within the network using the packet
18 send/receive module 1307.

19 The invention as described here can also be used to
20 check transformations that are only occurring on part
21 of the route on which IP packets flow. This can occur
22 in the cases where an IP communication tunnel is
23 established between two network routers, but the
24 validation programs are executing on the end-stations
25 between which packets are flowing. Figure 14 shows
26 such an environment. It shows two hosts 1401 and 1403
27 which are communicating with each other. An IP
28 communication tunnel is established only on a part of
29 the route between the hosts 1401 and 1403. In Figure
30 14, the partial transformations are done on the packets
31 between a router 1405 and the host 1403 only. The

1 router 1405 connects to the host 1401 via an network
2 1407 on which transformations do not take place, and is
3 referred to as the untransformed network. The packets
4 are transformed between the router 1405 and the host
5 1403, this part of the route is in the transformation
6 network 1409.

7 In order to validate that the transformations are
8 occurring properly in this configuration, the
9 validation client 1411 configures a filtering agent
10 1413 to become active inside the router 1405. The
11 filtering agent 1413 will capture a subset of packets
12 passing through the router and provide a copy of them
13 to the validation client 1411. The filters would be
14 set in a way so as to enable the filtering agent to
15 capture the subset of packets for which the validation
16 client is interested in validating that the
17 transformations do occur. An example of a scheme that
18 can be used is by the validation client 1411 marking
19 packets that leave the host 1401. The marking may be
20 done by setting a predetermined value into the Type Of
21 Service field within the IP packet header. Such a
22 field is defined by the Internet Protocol standards
23 known to those familiar with the art. The validation
24 client can then generate packets with specific markings
25 so that they would be captured by the filtering agent
26 1413. The filtering agent 1413 captures the packets
27 after their transformation and provides a copy of the
28 same to the validation client 1411. The validation
29 client 1411 verifies that the packets have been
30 transformed in the correct fashion.

1 Figure 15 shows a flowchart illustrating a method for
2 validation partial route transformations in the
3 environment demonstrated by Figure 14. The method is
4 entered by a validation client in the initialization
5 step 1501. In the next step 1503, the validation
6 client configures the sending device so that the
7 packets generated for the testing purposes would be
8 created with a specific Type of Service (TOS) marking.
9 One of the ways to implement step 1503 is by making a
10 function call to set TOS marking on the socket which
11 will be used to send the packets route. In step 1505,
12 the method configures the filtering agent on the router
13 to capture and send a copy of the packet back to the
14 validation client for the purpose of examination. In
15 step 1507, the validation client sends the a packet
16 out so that it would be marked with the desired TOS
17 marking. In step 1509, it receives a copy of the
18 transformed packet as obtained by the filtering agent.
19 If a packet is not received within a predetermined time
20 period, the method fails and stops in step 1511. If a
21 packet si receives, it checks in step 1513 if the
22 transformations have occurred correctly. If so, the
23 method terminates with success in step 1515, otherwise
24 it fails and terminates in step 1511.

25 It is noted that there are many variations of the
26 method illustrated in Figure 9 which can be used to
27 achieve the same results. One variant executes the
28 method illustrated in Figure 9, wherein the original
29 TTL value created in step 902 is selected so that the
30 TTL expires when the packet leaves the outbound router
31 1405 shown in Figure 14. Upon receiving an ICMP

1 message generated due to the expiry of TTL, the
2 validation client can check if the contents of the
3 packet have been transformed correctly. It is noted
4 that those familiar with the art will realize that the
5 apparatus for validating the transformation in a
6 partial route may be the same as those shown in Figures
7 12 and 13, wherein the transformation validator
8 implements the method shown in Fig 15.

9 The present invention can be realized in hardware,
10 software, or a combination of hardware and software. A
11 visualization tool according to the present invention
12 can be realized in a centralized fashion in one
13 computer system, or in a distributed fashion where
14 different elements are spread across several
15 interconnected computer systems. Any kind of computer
16 system - or other apparatus adapted for carrying out
17 the methods described herein - is suitable. A typical
18 combination of hardware and software could be a general
19 purpose computer system with a computer program that,
20 when being loaded and executed, controls the computer
21 system such that it carries out the methods described
22 herein. The present invention can also be embedded in
23 a computer program product, which comprises all the
24 features enabling the implementation of the methods
25 described herein, and which - when loaded in a computer
26 system - is able to carry out these methods.

27 Computer program means or computer program in the
28 present context include any expression, in any
29 language, code or notation, of a set of instructions
30 intended to cause a system having an information

1 processing capability to perform a particular function
2 either directly or after either or both of the
3 following a) conversion to another language, code or
4 notation; b) reproduction in a different material form.

5 It is noted that the foregoing has outlined some of the
6 more pertinent objects and embodiments of the present
7 invention. This invention may be used for many
8 applications. Thus, although the description is made
9 for particular arrangements and methods, the intent and
10 concept of the invention is suitable and applicable to
11 other arrangements and applications. It will be clear
12 to those skilled in the art that modifications to the
13 disclosed embodiments can be effected without departing
14 from the spirit and scope of the invention. The
15 described embodiments ought to be construed to be
16 merely illustrative of some of the more prominent
17 features and applications of the invention. Other
18 beneficial results can be realized by applying the
19 disclosed invention in a different manner or modifying
20 the invention in ways known to those familiar with the
21 art.